# c++primer 学习日记2 复合类型 引用与指针

本篇是反思后的第一篇日记，本系列，作为自己的c++primer的学习，c++的复习。

复合类型：引用，指针；

术语“引用（reference）”指左值引用；

引用即别名，例：

Int a=0;

Int &b=a;

引用与对象绑定，对引用的操作即为对对象的操作；

因为引用不是对象，不能定义引用的引用与引用的指针；

引用应与绑定对象严格匹配，且不能绑定字面值与某个表达式的计算结果；

“指针”与引用皆实现了对对象的间接访问，但其本身也是一个对象，允许对其赋值与拷贝；

指针与其对象类型也需严格匹配；

指针的值（即地址）应属于下列4种状态：

1 指向一个对象；

2指向紧邻对象的下一个位置；

3 空指针，不指向任何一个位置；

4无效指针，上述三种情况之外的指针；

获取地址 取地址符&；

如果指针指向一个对象，允许使用解引用符\*来访问该对象；

例：

 Int ival=42;

Int\*p=&ival;

Cout<<\*p;

对\*p赋值实际上是为p所指的对象赋值；

声明语句中 \*代表指针，&代表引。表达式中\*代表解引用，，&代表取地址。

空指针不指向任何对象

生成空指针方法：

Int  \*p1=nullptr;

Int\*p2=0；

Int\*p3=NULL；

其中nullptr是c++11新引入方法，是一种特殊类型的字面值，可以被转换为任意其他的指针类型。

int变量直接赋给指针是错误的操作，即便其值为零。

引用与指针同为间接访问对象，其区别在与引用绑定与一个对象，其本身不是对象。

赋值改变的永远是等号左侧的对象，等号左侧为p时，改变的是指针指向，左侧为p\*时，改变的是p指向的对象。

Void指针用于存放各种对象地址，

例，int a=3,\*b=&a;void\*c=&a;c=b;